Remarks

Entry of the above-noted amendments, reconsideration of the application, and allowance of all claims pending are respectfully requested. By this amendment, claims 1, 7 and 33 are amended. These amendments to the claims constitute a bona fide attempt by applicants to advance prosecution of the application and obtain allowance of certain claims, and are in no way meant to acquiesce to the substance of the rejections. Support for the amendments can be found throughout the specification (e.g., page 2, lines 15-19; page 3, lines 2-7), and drawings (e.g., FIGS. 2-3). Claims 1- 41 are pending.

Claim Rejections - 35 U.S.C. §103

MPEP §706.02(j) states: "To establish a prima facie case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art and not based on applicant's disclosure. In re Vaeck, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991)."

MPEP §2143.01 states: "Obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either explicitly or implicitly in the references themselves or in the knowledge generally available to one of ordinary skill in the art. The test for an implicit showing is what the combined teachings, knowledge of one of ordinary skill in the art, and the nature of the problem to be solved, as a whole would have suggested to those of ordinary skill in the art. In re Kotzab, 217 F.3d 1365, 1370, 55 USPQ2d 1313, 1317 (Fed. Cir. 2000). See also In re Fine, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988); In re Jones, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992)."

The Examiner rejected claims 1-41 under 35 U.S.C. 103 as being obvious based on cited pages of a book by Jaworski (Java 1.1 Developer's Guide Second Edition, 1997) in view of Venners' article at http://www.javaworld.com/javaworld/jw-05-1997/jw-05-hood _p.html. Applicants respectfully submit that the applied combination of references, assuming, *arguendo* that the modification or combination of the applied references is proper, does not teach or suggest one or more elements of the claimed invention, as further discussed below. Applicant respectfully traverses the rejections and seeks withdrawal of the rejections resulting in allowance of the application.

In claim 1, a software agent is generated at an originating host. The software agent is split into a code unit and a data unit. The data unit, but not the associated code unit, is forwarded from the originating host to the destination host. In the rejection of claim 1, the Venners reference was cited for generally teaching the generation of a software agent, i.e. aglets as Java objects. It was acknowledged that the Venners reference did not teach splitting the software agent into a code unit and a data unit. Jaworski was alleged to teach the splitting requirement and the forwarding of the data unit to a destination host. Applicant respectfully traverses this rejection since Jaworski does not teach splitting of the code unit and data unit, and forwarding the data unit but not the code unit to the destination host as explained below.

Pages 90-95 of Jaworski discuss various concepts of object-oriented programming. The following quotation from this section defines an "object" and should be considered:

> "Objects are defined in terms of the information they contain and the operations they provide for using and manipulating this information."; p. 90.

It is explained on page 93 that inheritance refers to the fact that a lower-level category of objects share the characteristics of the categories of objects above it on the classification tree. As pointed out above, objects are defined as containing information (data) and operations for manipulating the information (code).

Pages 124-126 are generally directed to describing Java interfaces. It is explained that interfaces provide a standard framework for accessing classes. In discussing the various characteristics of Java interfaces, it is explained:

> "Interfaces also support selective multiple inheritance. They allow various subsets of the features supported by different classes to be shared without mandating that all features of these classes be uniformly imposed as the result of inheritance."; p. 125.

The above quoted text from p. 125, which was relied upon in the Office Action as teaching the splitting of a software agent into a code unit and a data unit as per claim 1, does not provide the teaching attributed to it in the Office Action. The quoted text merely explains the principle of selective multiple inheritance with regard to classes of Java objects. It explains that interfaces allow some of the features (attributes or characteristics) supported by a class of objects to be shared without requiring that all of the features of a class of objects be shared. It does not teach or suggest the splitting of a single object to separate the information (data) and operations (code) contained within it. This text must be read in context in which a description is being provided of the concept of Java interfaces, i.e. the framework for accessing classes. It is the "features" of classes that is being explained. Thus, one of ordinary skill in the art would not be lead based on this explanation to consider splitting the data and code units of a software agent.

The discussion of "Encapsulation" on page 95 was also referenced in the Office Action with regard to the combining of data and code to form a single component, i.e. an object. This section reads:

> One characteristic of object-oriented programming that is often touted is encapsulation. The term carries the connotation of an object being enclosed in some sort of container-and that is exactly what it means. Encapsulation is the combining of data and code that manipulates that data into a single component-that is, an object. The encapsulation also refers to the control of access to the details of an object's implementation. Object access is limited to a well-defined, controlled interface. This allows objects to be self-contained and protects them from accidental misuse, both of which are important to reliable software design. The encapsulation provides the basis for developing security-rich objects.

The above quoted text merely explains that a Java object that contains information and operations to manipulate the information can be conceptually considered as an encapsulation, i.e. information and operations contained together. It should be noted that the object is still described as containing data and code that manipulates the data. This is consistent with the definition previously offered on page 90; see the text quoted above.

"An applicant may rebut a prima facie case of obviousness by showing that the prior art teaches away from the claimed invention in any material respect. In re Geisler, 116 F.3d at 1469, 43 USPQ2d at 1365 (quoting In re Malagari, 499 F.2d at 1303, 182 USPQ at 553)."

The basic definition of an object as containing data and code that manipulates the data is antithetical to the requirements of claim 1 which requires that the software agent be split into a separate code unit and data unit. The concept of encapsulation, referring to the basic nature of an object, even further teaches away from the splitting of a software agent into a separate code unit and data unit. The relied upon pages of Jaworski would not lead one of ordinary skill the art to consider splitting a software agent into separate code and data units. In fact, Jaworski explains that a single object contains both information and operations to act on the contained information. This does not support the requirements of claim 1, and in fact teaches away from the requirement of a software agent having a separated data and code unit.

Claim 1 is amended to further clarify and make explicit that the originating host forwards the data unit but not the associated code unit to the destination host. Neither Venners, Jaworski, nor the combination thereof renders the requirements of claim 1 obvious. Thus, it is submitted that claim 1 is allowable in view of the applied references.

Claim 4 further defines the step of combining the data unit with a watermark prior to the forwarding of the data unit to the destination host. This claim was rejected as "such features are well-known in the art for the motivation of security." Applicant respectfully traverses this rejection as failing to satisfy prima facie grounds for a valid rejection. Assuming for purposes of argument that use of a "watermark" is generally known, this does not make the combining of a data unit with a watermark prior to forwarding the data unit to the destination host obvious. In

the plurality of patents that are issued each Tuesday by the United States Patent and Trademark Office, almost all of the individual elements required in the many claims are known per se. It is the combination of elements and limitations that must be considered as a whole with regard to considering patentability, i.e. obviousness. The combination of separating the software agent into a code unit and a data unit, and then applying a watermark to the data unit prior to it being transmitted to the destination host that must be found to be obvious. The withdrawal of the rejection is sought. Should the rejection be maintained, Applicant requests the citation of prior art that teaches all of the required limitations as well as an appropriate motivation for making any required combinations.

Claim 7 describes the receiving of a data unit at a destination host from an originating host. The data unit is verified. The data unit is combined at the destination host with a code unit to form a software agent. The code unit as utilized by the destination host has not been received from the originating host. The destination host executes the software agent. Claim 7 was rejected based on Venners in view of Jaworski. It was acknowledged that Venners did not teach the receiving and combining steps in accordance with claim 7. The Jaworski was alleged to teach "receiving a data unit from an originating host; verified the data unit; combining the data unit with a code unit ..." based on the same pages alleged to support the rejection of claim 1. In considering the analogy of the software agent of claim 7 to Jaworski's teachings, the software agent would correspond to a Java object. As explained above Jaworski teaches that a Java object consists of both a data portion and an operation portion for manipulating the data. The Examiner has not pointed to a specific teaching in Jaworski in which a data unit alone has been received from an external source and combined with a code unit that was not received from the same external source. The above explanation provided with regard to claim 1 as to the teachings of Jaworski is also relevant to this rejection and are incorporated herein. The withdrawal of the rejection of claim 7 is requested.

Claim 10 further defines the method of claim 7 in that the code unit is obtained from an external source. That is, the code unit is obtained outside of the destination host from a source other than the origination host. Claim 10 was rejected the same reasons given for claim 7 and further in view that "such features are well-known in the art for the motivation of security."

Applicant respectfully traverses this rejection as not providing a valid prima facie ground for rejection. Applicant respectfully requests citation of appropriate prior art showing such a requirement if the rejection is maintained.

The rejection of claim 15 is traversed. The reasons explained with regard to claim 4 are incorporated herein.

The rejection of independent claim 19 is traversed for failure to state a valid prima facie ground of rejection. Claims 8-32 were summarily rejected based on "various data handlings, especially in encryption and identification-such features are well-known in the art for the motivation of security." Claim 19 describes a method, part of which is implemented by an originating host and another part implemented by a destination host. It is respectfully submitted that the required steps in accordance with claim 19 are not "well-known in the art". It is clear that the MPEP requires an appropriate application of prior art in support of a rejection as being obvious, especially with regard to rejecting an independent claim. Applicant is not required to guess what portions or combinations of one or more references may have been intended as grounds for supporting the rejection.

Pursuant to MPEP 706.07(c), it would be inappropriate to make an Office Action final should new references or grounds not previously stated be applied in support of a rejection of any unamended claims, e.g. claim 19, where applicant has made no amendment to necessitate such a change of position.

Independent apparatus claim 33 was rejected as being obvious based on Venners in view of Jaworski for reasons similar to the reasons expressed with regard to the rejection of claims 1 and 7. Applicant respectfully traverses this rejection for the same reasons expressed above with regard to claims 1 and 7. Withdrawal of the rejection is sought.

In view of the above amendments and remarks, allowance of all claims pending is respectfully requested. If a telephone conference would be of assistance in advancing the prosecution of this application, the Examiner is invited to call applicants' attorney.

Respectfully submitted,

Charles L. Warren
Attorney for Applicants
Reg. No. 27,407

Dated: December 14, 2004

PATTI & BRILL, LLC
Customer Number 32205